链滴

STM32 IWDG 独立看门狗

作者：Today

原文链接：https://ld246.com/article/1528523776631

来源网站：链滴

STM32 独立看门狗简介

在学习STM32的看门狗的之前要先了解什么是** 看门狗 **。

看门狗定时器（WDT，Watch Dog Timer）是单片机的一个组成部分，它实际上是一个计数器，一给看门狗一个数字，程序开始运行后看门狗开始倒计数。如果程序运行正常，过一段时间CPU应发出令让看门狗复位 ，重新开始倒计数。如果看门狗减到0就认为程序没有正常工作，强制整个系复位。

在我看来看门狗就是用来监听程序是否正常执行，如果不正常就会整个程序就会自动复位。人（程序定时喂狗，当喂狗的人（程序）死了，狗为了活命于是救活了（程序）人,人(程序)复活后又继续喂狗。

STM32 的独立看门狗由内部专门的 40Khz 低速时钟驱动,即使主时钟发生故障,它也仍然有效。这里要注意独立看门狗的时钟是一个内部 RC 时钟,所以并不是准确的 40Khz,而是在 30~60Khz 之间的一可变化的时钟,只是我们在估算的时候,以 40Khz 的频率来计算,看门狗对时间的要求不是很精确,所以,钟有些偏差,都是可以接受的。

独立看门狗有几个主要的寄存器：

1.键值寄存器（IWDG_KR)

低16位寄存器。软件必须每隔一定时间写入**0xAAAA**,否则，当计数器减到0时，程序会自动复位。

- 写入 **0x5555**表示允许访问**IWDG_PR**和**IWDG_RLR**寄存器。
- 写入 **0xCCCC**时启动看门狗工作（若选择了硬件则不受此命令限制）。

在键寄存器(**IWDG_KR**)中写入**0xCCCC**,开始启用独立看门狗，此时计数器开始从其复位值 **0xFFF** 递计数。当计数器计数到末尾 **0x000 （一般不会为0为0时程序跑飞）时,会产生一个复位信号(WDG_RESET)。无论何时,只要键值寄存器 IWDG_KR 中被写入 0xAAAA, IWDG_RLR 中的值就会被重加载到计数器中从而避免产生看门狗复位（程序运行正常）** 。

2.预分频寄存器(IWDG_PR)

该寄存器用来设置看门狗时钟的分频系数,最低为 **4（000）**,最高位 **256（111）**,该寄存器是一个 **32** 的寄存器,但是只用最低 **3** 位

3.重装载寄存器（IWDG_RLR)

更具选择的预分频和最长喂狗时间设置重装载寄存器值**（**最长喂狗时间=((4×2^预分系数) ×重装载值) /40 ms**）。**现在假设我们必在1秒内喂狗否则就认为程序跑飞。当预分频系数为4时**（预分频因子为64）**根据上面的公式可以得重装值为**：625**

iwdg.h

```
#ifndef IWDG_IWDG_H_
#define IWDG_IWDG_H_

#include <stm32f10x.h>

void IWDG_INIT(uint32_t prer, uint32_t rlr);
void IWDG_INIT_REG(uint32_t IWDG_Prescaler, uint32_t Reload_Value);

#endif /* IWDG_IWDG_H_ */
```

iwdg.c

```
#include "iwdg.h"


#include <stm32f10x_iwdg.h>
#include <sys/_stdint.h>

void IWDG_INIT(u8 IWDG_Prescaler, u16 Reload) {

    IWDG_WriteAccessCmd(IWDG_WriteAccess_Enable);//取消寄存器写保护
    IWDG_SetPrescaler(IWDG_Prescaler);//预分频寄存器
    IWDG_SetReload(Reload);
    IWDG_ReloadCounter(); //从重装值加载
    IWDG_Enable();
}

//最长喂狗时间=((4×2^预分频系数) ×重装载值) /40 ms
void IWDG_INIT_REG(uint32_t IWDG_Prescaler, uint32_t Reload_Value){

    IWDG->KR = IWDG_WriteAccess_Enable;  //键值寄存器 开始启用独立看门狗
    IWDG->PR = IWDG_Prescaler;           //预分频寄存器
    IWDG->RLR = Reload_Value;            //重装载寄存器 重新装载
    IWDG->KR = 0xaaaa;                  //oxaaaa
    IWDG->KR = 0xcccc;
}
```

main.c

```
#include <stm32f10x.h>
```

```
#include <stm32f10x_gpio.h>

#include <stm32f10x_iwdg.h>

#include  <stm32f10x_rcc.h>



#include "delay/delay.h"
#include "iwdg/iwdg.h"
#include "led/led.h"

#define LED_0 GPIO_Pin_15
#define KEY_0 GPIO_Pin_0

//初始化LED
void LED_Init(){

    GPIO_InitTypeDef GPIO_Init_type;

    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB, ENABLE);
    GPIO_Init_type.GPIO_Mode          = GPIO_Mode_Out_PP;
    GPIO_Init_type.GPIO_Pin           = GPIO_Pin_All;
    GPIO_Init_type.GPIO_Speed         = GPIO_Speed_50MHz;
    GPIO_Init(GPIOB, &GPIO_Init_type);
    GPIO_SetBits(GPIOB, GPIO_Pin_All);

    GPIO_Init_type.GPIO_Mode        = GPIO_Mode_IPU;
    GPIO_Init_type.GPIO_Pin         = KEY_0;
    GPIO_Init(GPIOB, &GPIO_Init_type);
}

int main(int argc, char **argv) {

    delay_init();
    LED_Init();
    delay_ms(200);
    GPIO_WriteBit(GPIOB, GPIO_Pin_15, (BitAction) 0);
    IWDG_INIT(IWDG_Prescaler_64, 625); //1s喂一次 最长喂狗时间=( 64 ×重装载值) /40 ms

    while (1)
    {
        if (!GPIO_ReadInputDataBit(GPIOB, KEY_0)){
            delay_ms(20);
            if (!GPIO_ReadInputDataBit(GPIOB, KEY_0)){
                IWDG_ReloadCounter();//向键值寄存器写0xAAAA 重装载操作
            }
        }
    }

}
```

程序效果：如果不按下按键LED会每隔1200ms闪烁一次，如果在LED亮着的时候按下按键LED会

直亮着。